STAC BENCHMARK COUNCIL

Overview of the STAC-M3™ Benchmark Specifications

(Antuco Suite)

Analyzing time-series data such as tick-by-tick quote and trade histories is crucial to many trading functions, from algorithm development to risk management. But the domination of liquid markets by automated trading—especially high-frequency trading—has made such analysis both more urgent and more challenging. As trading robots try to outwit each other on a microsecond scale, they dish out quotes and trades in ever more impressive volumes. This places a premium on technology that can store and analyze that activity efficiently. For example, the faster an algorithm developer can back-test and discard a haystack of unprofitable ideas, the faster he will find the needle of a winning algorithm, leaving more time to exploit it in the market.

The STAC Benchmark Council has developed the STAC-M3 Benchmarks in order to provide a common basis for quantifying the extent to which emerging hardware and software innovations improve the performance of tick storage, retrieval, and analysis.

STAC-M3 tests the ability of a solution stack such as columnar database software, servers, and storage, to perform a variety of operations on a large store of market data. The STAC-M3 Working Group designed these test specs to enable useful comparisons of entire solution stacks (i.e., to gauge the state of the art) as well as comparisons of specific stack layers while holding other layers constant. Comparisons can include (but are not limited to):

- Different storage systems, including SSD, DRAM, interconnects, and file systems
- Different server products, processors, chipsets, and memory
- Different tick-database products

The following pages provide a high-level overview of the test configuration, dataset, metrics, and test cases used in STAC-M3.

Get the most from STAC-M3

Any interested party can analyze public STAC Reports to compare the performance of different systems. However, members of the STAC Benchmark Council are able to put these reports to much greater use. Qualified members may:

- Read the detailed test specifications
- Access additional reports in the confidential STAC Vault™
- Obtain the materials to run the STAC-M3 Benchmarks on their own systems
- Discuss benchmarks, technologies, and related business issues with their peers.

For more information, please contact council@STACresearch.com.

Test configuration

As Figure 1 shows, the test setup for STAC-M3 consists of the "stack under test" (SUT) and client applications. No restrictions are placed on the architecture of the SUT or clients (though members of the STAC-M3 Working Group frequently provide input on architectures they would like to see tested). Threads within the clients take in Randomized Reference Data (RRD) such as dates and symbols, submit requests for the required operations, receive responses, and store the timings and results from these queries. Vendor-supplied code for the operations and latency calculations are subjected to a combination of source-code inspection and empirical validation.

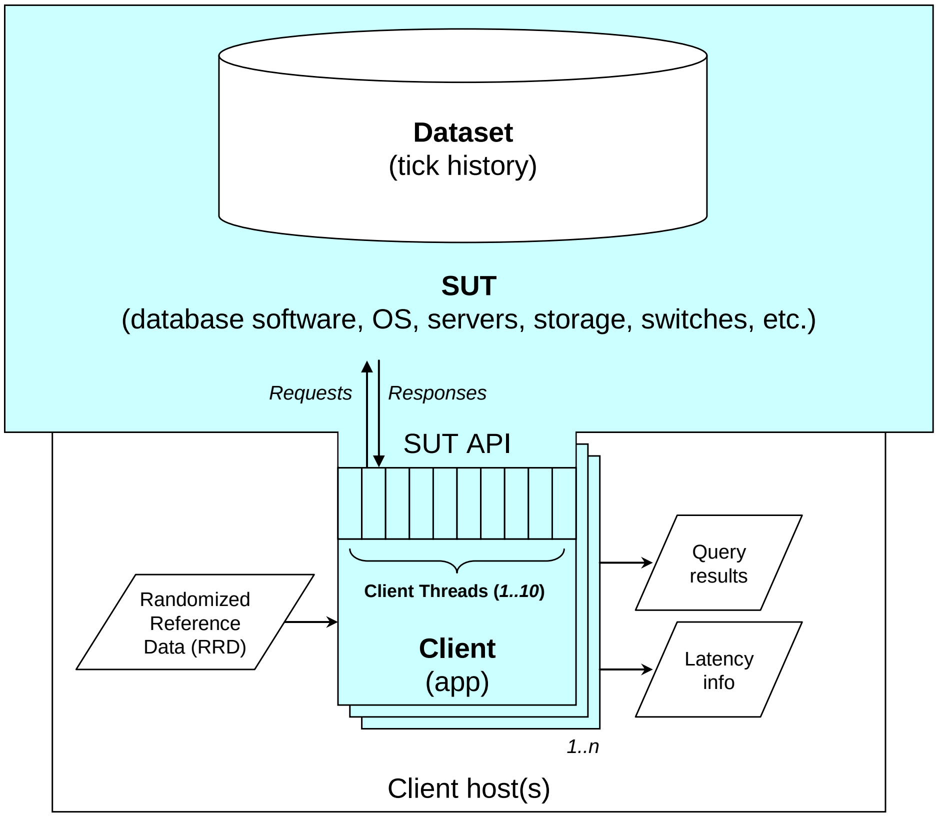


Figure 1 – Test configuration

Dataset

STAC-M3 draws from client experience with equities and FX use cases. The database is synthetic, modeled on NSYE TAQ data (US equities). While it is also desirable to test with real data, synthetic data has three advantages that make it compelling for this STAC-M3 suite:

- Synthetic data allows us to control the database properties exactly, which in turn allows us to randomize elements of queries from project to project while keeping the resulting workload exactly the same (for example, we control how much volume is associated with each symbol).
- Synthetic data does not incur fee liability from a third party such as an exchange.
- Synthesizing the data makes it easy to scale the database to an arbitrarily large size and run benchmarks against projected future data volumes.

The dataset consists of high-volume symbols and low-volume symbols in proportions based on observed NYSE data. The data volume per symbol was based on doubling the typical volume in NYSE TAQ in 1Q10. The resulting database is considerably smaller than databases in use at customer sites. This was a deliberate choice by the STAC-M3 Working Group to minimize the cost of running benchmarks while still yielding valuable results. Benchmarks that scale the database to the size of existing customer footprints and well beyond are under consideration for a future STAC-M3 suite.

Metrics

The key metric in STAC-M3 is the latency of query responses (aka response times). Latency measurements are performed in the clients. A client thread gets a local timestamp (t_{submit}) just before submitting a query. When the first results arrive, the client gets another timestamp (t_{first}). When it receives the complete results (sorted appropriately), the client immediately gets a third timestamp (t_{last}). For systems that return all results in one chunk, the first-result and last-result timestamp are identical. As Figure 8 illustrates, latencies are defined as follows:

$$\text{First-result latency (LAT1)} = t_{first} – t_{submit}$$

$$\text{Last-result latency (LAT2)} = t_{last} – t_{submit}$$

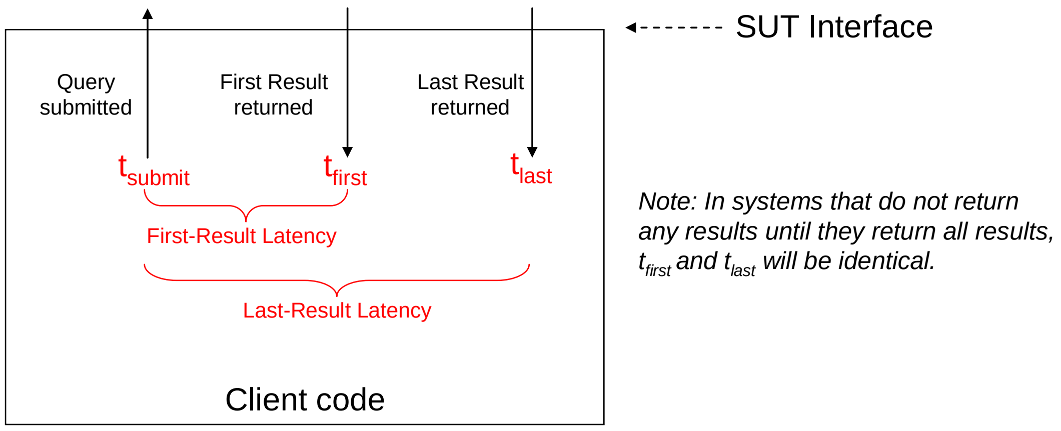


Figure 2 – Timestamp and latency meanings

The algorithms in all benchmarks are defined so as to keep the result sets small. This ensures that network I/O between the test clients and server(s) is negligible compared to back-end processing times.

Some of the I/O-focused benchmarks also measure the bytes read per second from persistent storage (i.e., excluding server cache), which is computed from the output of appropriate system utilities.

Test cases

The current tests in the STAC-M3 suite are listed in the Summary Table below. As the versioning illustrates, the STAC-M3 Working Group is bringing benchmark specifications to market in phases. The first set of approved specs (those marked as “v1” or, in one case, “v1.1”) focused on storage-system performance with respect to heavy historical data loads. These workloads were deliberately light on compute and heavy on I/O. The second phase, (the Antuco suite) added benchmark specs that involve more compute-intensive analytics. These new specifications, marked as “ß1”, have not been put to a vote by the full STAC Benchmark Council and will become v1 specs if approved. These new benchmarks operate by symbol on many fields of underlying tick data for both trades and quotes across varying time windows.[1] The table classifies each test case as relatively heavy on I/O, compute, or both.

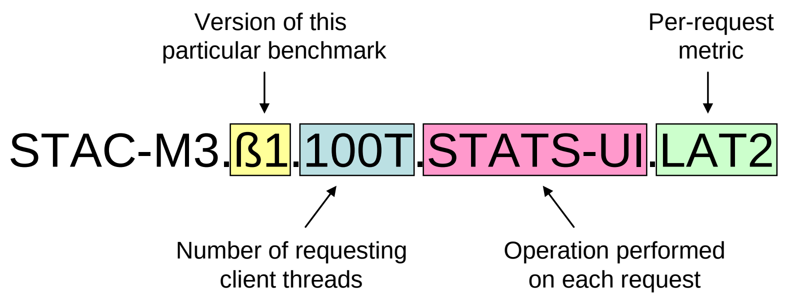
The tests require a client application that is written to a product API and is capable of submitting requests from 10 independent threads. As detailed in the table, some of the benchmarks call for one client instance making requests from a single thread, while others call for one client using 10 threads, and still others require 10 clients each using 10 threads (100 total requesting threads). One set of benchmarks (using the STATS-UI operation) tests multi-user scaling by running with 1, 10, 50, and 100 client threads and allowing the tester to scale to even higher numbers of concurrent threads. In all cases, benchmark

[1] Future phases will supplement these with benchmarks that provide insight into additional aspects of system performance. Contact council@STACresearch.com if you would like to be part of the process.

results refer to per-request latency. For example, the mean of 10T.MKTSNAP.LAT2 is the mean time to satisfy a single market-snapshot request, not the total time to satisfy requests from all 10 client threads.

The range of dates eligible for querying depends on the benchmark. For example, some algorithms operate on dates randomly chosen throughout the year, some stick to a recent date range, and some always run on the most recent date (see the “Input Date Range” column of the table). The purpose of this is to provide a realistic optimization strategy for systems with multiple storage tiers of different speed, such as solid-state disk (SSD) and spindle-based storage. For example, suppose the tester wanted to demonstrate the benefit of adding an SSD to a system that otherwise relied on spindle-based storage. Furthermore, suppose the SSD was only large enough to store 1/5 of the test database. Benchmarks that accessed dates throughout the entire database would show some performance improvement when the SSD is added, since 1/5 of the queries would enjoy acceleration. In the real world, however, customers don’t simply add faster storage and randomly allocate data to it. Rather, they typically reserve the fastest storage tier for the most frequently accessed data or data accessed by especially time-sensitive algorithms. In STAC-M3, the tester can load the most recent data into the fastest storage. In the case above, the benchmarks that operate on, say, the most recent day (1/252 of the database) or the most recent month (1/12 of the database) will show the maximum improvement possible from the SSD, since their entire queries can be satisfied from SSD. This provides the clearest indication of acceleration possibilities for datasets that can fit within the faster tier.

In STAC-M3 reports, all tables and charts identify each benchmark unambiguously, as follows:


Version of this particular benchmark
Per-request metric
STAC-M3.ß1.100T.STATS-UI.LAT2
Number of requesting client threads
Operation performed on each request

In charts, the ID is sometimes decomposed, with part of it in the chart title or labels. Each individual STAC Benchmark™ specification has its own version number. The same version of a given spec may appear in multiple benchmark suites. Thus, the code names of the suites are irrelevant when making comparisons. Versioning individual specs enables the reader to compare a discrete result from one “stack under test” (SUT) to the corresponding result from another SUT. When making comparisons, be sure that the identifiers match exactly. If they do not, the benchmark results cannot be fairly compared.

Summary Table – STAC-M3 Benchmarks in the Antuco Suite

The table below gives a brief overview of each test in the latest STAC-M3 suite. Version numbers of 1 or greater indicate benchmark specs that have been approved. Versions less than 1 are proposed by the STAC-M3 Working Group but not yet voted on by the full STAC Benchmark Council.

Root ID	Operation name	Ver	Number of requesting Client Threads	Algorithm performed on behalf of each requesting Client Thread	Algorithm I/O intensity	Algorithm compute intensity	Input date range*
VWAB-D	VWAB-Day	1	1	4-hour volume-weighted bid over one day for 1% of symbols (like VWAP but operating on quote data, so much higher input volume).	Heavy read	Light	Last 30 days
VWAB-12D-NO	VWAB-12DaysNoOverlap	1	100	4-hour volume-weighted bid over 12 days for 1% of symbols. No overlap in symbols among client threads.	Heavy read	Light	Full year
YRHIBID	Year High Bid	ß1	1	Max bid over the year for 1% of symbols.	Heavy read	Light	Full year
YRHIBID-2	Year High Bid Re-run	ß1	1	Re-run of YRHIBID (same symbols) without clearing the cache.	Heavy read†	Light	Full year
QTRHIBID	Quarter HighBid	ß1	1	Max bid over the quarter for 1% of symbols.	Heavy read	Light	Most recent quarter
MOHIBID	Month High Bid	ß1	1	Max bid over the month for 1% of symbols.	Heavy read	Light	Most recent month
WKHIBID	Week High Bid	ß1	1	Max bid over the week for 1% of symbols.	Heavy read	Light	Most recent week
STATS-AGG	Aggregate Stats	ß1	10	One set of basic statistics over 100 minutes for all symbols on one exchange. Each 100-minute range crosses a date boundary.	Heavy read	Heavy	Full year
STATS-UI	Stats - Unpredictable Intervals	ß1	1, 10, 50, 100 (more optional)	Per-minute‡ basic statistics over 100 minutes for all high-volume symbols on one exchange. Each 100-minute range crosses a date boundary.	Heavy read	Heavy	Full year

MKTSNAP	Market Snapshot	ß1	10	Most recent trade and quote information for 1% of symbols as of a random time.	Heavy read	Heavy	Full year
VOLCURV	Volume Curves	ß1	10	Create an average volume curve (using minute intervals aligned on minute boundaries) for 10% of symbols over 20 days selected at random.	Light read	Heavy	Full year
THEOPL	Theoretical P&L	ß1	10	For a basket of 100 trades on random dates, find the future times at which 2X, 4X, and 20X the trade size traded in each symbol. Trade sizes cause up to 5 days of forward searching. Calculate the corresponding VWAP and total volume traded over those periods.	Light read	Heavy	Full year
NBBO	NBBO	ß1	1	Create the NBBO across all 10 exchanges for all symbols on the most recent day. Write to persistent storage.	Heavy read and write	Heavy	Most recent day
WRITE	Write	1	1	Write one day's quote data to persistent storage, following the same algorithm used to generate the randomized dataset used in the other Operations.	Heavy write	Light	n/a
STORAGE.EFF	Storage efficiency	1.1	n/a	Reference Size of the Dataset divided by size of the Dataset in the SUT format used for the performance benchmarks. Expressed as as percentage.	n/a	n/a	n/a

[*] In some cases, one or more dates at the end of the year were excluded from eligibility to prevent an algorithm that crosses days from running out of input data.
[†] Typically this will be reads from DRAM cache.
[‡] In this case, interval start times are offset from minute boundaries by a consistent random amount per test run, so that the SUT cannot rely on pre-calculated minute statistics.